AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application:

LISTING OF CLAIMS:

1. (currently amended): A networked system comprising:

a host computer;

a data streamer connected to said host computer, said data streamer capable of transferring data between said host and networked resources using a memory ~~location~~ within said data streamer without moving the data ~~within the~~ between memory locations within the memory during processing by said data streamer;

a communication link connecting said data streamer and networked resources.

2. (original): The system of claim 1, wherein said communication link is a dedicated communication link.

3. (currently amended): The system of claim 1, wherein said host computer is used solely for initializing the ~~computer~~ networked system.

4. (original): The system of claim 1, wherein the networked resources include networked storage devices.

5. (original): The system of claim 2, wherein the dedicated communication link is a network communication link.

6. (original): The system of claim 3, wherein the dedicated communication link is selected from a group consisting of personal computer interface (PCI), PCI-X, 3GIO, InfiniBand, SPI-3, or SPI-4.

7. (currently amended): The system of claim 5, wherein the network communication link is at least one of:

a local area network (LAN) link[[.]], a wide area network (WAN).

8. (currently amended): The system of claim 5, wherein the network communication link is ~~Ethernet~~ based~~.~~ on at least one of:

Ethernet, Internet protocol (IP), asynchronous transfer mode (ATM) protocol.

9. (currently amended): The system of claim 1~~5~~, wherein said data streamer is configured to relieve said host from at least upper level protocol (ULP) processing ~~the network communication link is a wide area network (WAN).~~

10.-11. canceled.

12. (original): The system of claim 1, wherein said data streamer further comprises:

at least one host interface, interfacing with said host computer;

at least one network interface, interfacing with the networked resources;

at least one processing node, capable of generating additional data and commands necessary for network layer operations;

an admission and classification unit that initially processes the data;

an event queue manager that supports processing of the data;

a scheduler that supports processing of the data;

a memory manager that manages the memory;

a data interconnect unit that receives the data from said admission and classification unit; and

a control hub.

13. (original): The system of claim 12, wherein said processing node is further connected to an expansion memory.

14. (original): The system of claim 13, wherein said expansion memory is a code memory.

15. (original): The system of claim 12, wherein said processing node is a network event processing node.

16. (original): The system of claim 15, wherein said network event processing node is a packet processing node.

17. (original): The system of claim 15, wherein said network event processing node is a header processing node.

18. (original): The system of claim 12, wherein said host interface is selected from a group consisting of PCI, PCI-X, 3GIO, InfiniBand, SPI-3, and SPI-4.

19. (original): The system of claim 12, wherein the network interface is Ethernet.

20. (original): The system of claim 12, wherein the network interface is ATM.

21. (original): The system of claim 12, wherein said host interface is combined with the network interface.

22. (original): The system of claim 12, wherein said event queue manager is capable of managing at least:

an object queue; and

an application queue.

23. (currently amended): The system of claim 22, wherein said object queue points to a first descriptor while a first header is processed.

24. (currently amended): The system of claim 23, wherein ~~the~~ a header of the data processed is in one of: ~~the~~ a second communication layer~~.~~, a third communication layer, a fourth communication layer, and a fifth communication layer.

25.-26. canceled.

27. (currently amended): The system of claim 23, wherein said object queue points to a second descriptor if ~~the~~ a second header has ~~the~~ a same tuple corresponding to the first header.

28. (currently amended): The system of claim 22, wherein said object queue holds at least ~~the~~ a start address to the header information.

29. (currently amended): The system of claim 22, wherein said object queue hold at least ~~the~~ an end address to the header information.

30. (currently amended): The system of claim 23, wherein said application queue points to said first descriptor instead of said object queue if at least an application header is available.

31. (currently amended): The system of claim ~~23~~ 30, wherein said first descriptor points at least to ~~the~~ a beginning of the application header.

32. (currently amended): The system of claim 31, wherein said application queue maintains address of said beginning of the application header.

33. (currently amended): The system of claim 23, wherein said first descriptor points at least to ~~the~~ an end of said application header.

34. (currently amended): The system of claim 33, wherein said application queue maintains address of said end of said application header.

35. (currently amended): The system of claim 30, wherein when all ~~the~~ application headers are available, data is transferred to said host computer in a continuous operation.

36. (original): The system of claim 35, wherein said continuous operation is based on pointer information stored in said application queue.

37. (original): The system of claim 22, wherein the system is adapted to receive at least one packet of data with headers from a network resource and opening a new descriptor if the headers do not belong to a previously opened descriptor.

38. (currently amended): The system of claim 37, wherein the system is adapted to store ~~the~~ start and end address of the headers in the object queue.

39. (original): The system of claim 37, wherein the system is adapted to transfer control of the descriptor to the application queue if at least one application header is available and is further adapted to store a start and end address of the application header in the application queue.

40. (original): The system of claim 39, wherein the system is adapted to transfer the data to the host based on the stored application headers.

41. (original): The system of claim 22, wherein the system is adapted to receive data and a destination address from the host computer, and further wherein the system is adapted to queue the data in a transmission queue.

42. (original): The system of claim 41, wherein the system is adapted to update an earlier created descriptor to point to a portion of the data that is to be sent next.

43. (original): The system of claim 42, wherein the system is adapted to create headers and attach the portion of the data to the headers and transmit them over the network.

44. (currently amended): A data streamer for use in a network, said streamer comprising:

at least one host interface, interfacing with ~~said~~ a host computer;

at least one network interface, interfacing with ~~the~~ networked resources;

at least one processing node, capable of generating additional data and commands necessary for network layer operations;

an admission and classification unit that initially processes the data;

an event queue manager that supports processing of the data;

a scheduler that supports processing of the data;

a memory manager that manages ~~the~~ memory;

a data interconnect unit that receives the data from said admission and classification unit; and

a control hub~~.~~;

the data streamer configured to enable the relief of the host computer from at least upper level protocol (ULP) processing.

45. (original): The streamer of claim 44, wherein said processing node is further connected to an expansion memory.

46. (original): The streamer of claim 45, wherein said expansion memory is a code memory.

47. (original): The streamer of claim 44, wherein said processing node is a network event processing node.

48. (original): The streamer of claim 47, wherein said network event processing node is a packet processing node.

49. (original): The streamer of claim 47, wherein said network event processing node is a header processing node.

50. (original): The streamer of claim 44, wherein said host interface is selected from a group consisting of PCI, PCI-X, 3GIO, InfiniBand, SPI-3, and SPI-4.

51. (original): The streamer of claim 44, wherein the network interface is Ethernet.

52. (original): The streamer of claim 44, wherein the network interface is ATM.

53. (original): The streamer of claim 44, wherein said host interface is combined with the network interface.

54. (original): The streamer of claim 44, wherein said event queue manager is capable of managing at least:

an object queue;

an application queue.

55. (original): The streamer of claim 54, wherein said object queue points to a first descriptor while first header is processed.

56. (currently amended): The streamer of claim 55, wherein ~~the~~ a header of the data processed is in one of:

~~the~~ a second communication layer~~.~~, a third communication layer, a fourth communication layer, and a fifth communication layer.

57.-58. canceled.

59. (currently amended): The streamer of claim 55, wherein said object queue points to a second descriptor if ~~the~~a second header has ~~the~~ a same tuple corresponding to the first header.

60. (currently amended): The streamer of claim 54, wherein said object queue hold at least ~~the~~a start address to the header information.

61. (currently amended): The streamer of claim 54, wherein said object queue hold at least ~~the~~ an end address to the header information.

62. (currently amended): The streamer of claim 55, wherein said application queue points to said first descriptor instead of said object queue if at least an application header is available.

63. (currently amended): The streamer of claim ~~55~~ 62, wherein said first descriptor points at least to ~~the~~ a beginning of the application header.

64. (currently amended): The streamer of claim 63, wherein said application queue maintains address of said beginning of the application header.

65. (currently amended): The streamer of claim 55, wherein said first descriptor points at least to ~~the~~ an end of said application header.

66. (currently amended): The streamer of claim 65, wherein said application queue maintains address of said end of said application header.

67. (currently amended): The streamer of claim 62, wherein when all the application headers are available, data is transferred to said host computer in a continuous operation.

68. (original): The streamer of claim 67, wherein said continuous operation is based on pointer information stored in said application queue.

69. (original): The streamer of claim 54, wherein the streamer is adapted to receive at least one packet of data with headers from a network resource and opening a new descriptor if the headers do not belong to a previously opened descriptor.

70. (currently amended): The streamer of claim 69, wherein the streamer is adapted to store ~~the~~ start and end address of the headers in the object queue.

71. (original): The streamer of claim 70, wherein the streamer is adapted to transfer control of the descriptor to the application queue if at least one application header is available and is further adapted to store a start and end address of the application header in the application queue.

72. (original): The streamer of claim 71, wherein the streamer is adapted to transfer the data to the host based on the stored application headers.

73. (original): The streamer of claim 54, wherein the streamer is adapted to receive data and a destination address from the host computer, and further wherein the streamer is adapted to queue the data in a transmission queue.

74. (original): The streamer of claim 73, wherein the streamer is adapted to update an earlier created descriptor to point to a portion of the data that is to be sent next.

75. (original): The streamer of claim 74, wherein the streamer is adapted to create headers and attach the portion of the data to the headers and transmit them over the network.

76. (currently amended): A method for transferring application data from a network to a host computer comprising:

a) receiving headers of data from a network resource;

b) opening a new descriptor if the headers do not belong to a previously opened descriptor;

c) storing a start address and an end address of the headers in an object queue;

d) transferring control of the descriptor to an application queue if at least one application header is available;

e) storing start and end address of the application header in an application queue;

f) repeating steps a through e) until all application headers are available; and

g) transferring the data to said host based on said application headers.

77. (original): A method for transferring application data from a host computer to a network resource comprising:

a) receiving data from the host computer;

b) receiving destination address from the host computer;

c) queuing a transmission information in a transmission queue;

d) updating a descriptor pointing to portion of the application data to be sent next;

e) creating headers for the transmission;

f) attaching the portion of the application data to the headers;

g) transmitting the portion of the application data and headers over the network;

h) repeating steps d through g until all of the application data is sent; and

i) indicating to the host computer that transfer is complete.

78. (new): A networked system comprising:

a host computer;

a data streamer connected to said host computer,

the data streamer configured to relieve the host computer from at least upper level protocol (ULP) processing;

a communication link connecting said data streamer and networked resources.

79. (new): The networked system of claim 78, wherein the data streamer is further configured for transferring data between said host and networked resources using a memory within said data streamer without moving the data between memory locations within the memory during processing by said data streamer.